Supervised locally linear embedding

Dick de Ridder[1], Olga Kouropteva[2], Oleg Okun[2],
Matti Pietikäinen[2] and Robert P.W. Duin[1]

[1] Pattern Recognition Group, Department of Imaging Science and Technology,
Delft University of Technology, Lorentzweg 1, 2628 CJ Delft, The Netherlands
{dick,bob}@ph.tn.tudelft.nl http://www.ph.tn.tudelft.nl

[2] Machine Vision Group, Infotech Oulu and Department of Electrical and
Information Engineering, P.O.Box 4500, FIN-90014 University of Oulu, Finland
{kouropte,oleg,mkp}@ee.oulu.fi http://www.ee.oulu.fi/mvg/mvg.php

Abstract. Locally linear embedding (LLE) is a recently proposed method for unsupervised nonlinear dimensionality reduction. It has a number of attractive features: it does not require an iterative algorithm, and just a few parameters need to be set. Two extensions of LLE to supervised feature extraction were independently proposed by the authors of this paper. Here, both methods are unified in a common framework and applied to a number of benchmark data sets. Results show that they perform very well on high-dimensional data which exhibits a manifold structure.

1 Introduction

In many real-world classification problems, high-dimensional data sets are collected, e.g. from sensors. Often, the ideal decision boundary between different classes in such sets is highly nonlinear. A classifier should therefore have many degrees of freedom, and consequently a large number of parameters. As a result, training a classifier on such data sets is quite complicated: a large number of parameters has to be estimated using a limited number of samples. This is the well-known *curse of dimensionality*.

One can overcome this problem by first mapping the data to a high-dimensional space in which the classes become (approximately) linearly separable. Kernel-based techniques, such as support vector machines (SVMs), are typical examples of this approach. An alternative is to *lower* the data dimensionality, rather than increase it. Although it might seem information is lost, the reduction in the number of parameters one needs to estimate can result in better performance. Many linear methods for performing dimensionality reduction, such as principal component analysis (PCA) and Fisher's linear discriminant analysis (LDA) are well-established in literature.

Here, a nonlinear dimensionality reduction method called locally linear embedding (LLE, [8]) is considered. The main assumption behind LLE is that the data set is sampled from a (possibly nonlinear) manifold, embedded in the high-dimensional space. LLE is an unsupervised, non-iterative method, which avoids the local minima problems plaguing many competing methods (e.g. those based on the EM algorithm). Some other advantages of LLE are that few parameters need to be set (selecting optimal values for these is discussed in [3, 6]) and that the local geometry of high-dimensional data is preserved in the embedded space.

To extend the concept of LLE to multiple manifolds, each representing data of one specific class, two supervised variants of LLE were independently proposed in [3, 5]. In this paper, a framework unifying the unsupervised and both supervised methods is given. Supervised LLE is then applied as a feature extractor on a number of benchmark data sets, and is shown to be useful for high-dimensional data with a clear manifold structure.

2 LLE framework

2.1 LLE

As input, LLE takes a set of N D-dimensional vectors assembled in a matrix $\boldsymbol{X}$ of size $D \times N$. Its output is a set of N M-dimensional vectors ($M \ll D$) assembled in a matrix $\boldsymbol{Y}$ of size $M \times N$, where the k^{th} column vector of $\boldsymbol{Y}$ corresponds to the k^{th} column vector of $\boldsymbol{X}$. First, the $N \times N$ Euclidean distance matrix $\boldsymbol{\Delta}$ between all samples is constructed. For each sample $\boldsymbol{x}_i$, $i = 1, \ldots, N$, its K nearest neighbours are then sought; their indices are stored in an $N \times K$ matrix $\boldsymbol{\Gamma}$, such that Γ_{ij} is the index of the j-nearest neighbour of sample $\boldsymbol{x}_i$.

In the first step, each sample $\boldsymbol{x}_i$ is approximated by a weighted linear combination of its K nearest neighbours, making use of the assumption that neighbouring samples will lie on a locally linear patch of the nonlinear manifold. To find the reconstruction weight matrix $\boldsymbol{W}$, where $\boldsymbol{W}_{i\Gamma_{ij}}$ contains the weight of neighbour j in the reconstruction of sample $\boldsymbol{x}_i$, the following expression has to be minimised w.r.t. $\boldsymbol{W}$ [8]:

$$\varepsilon_I(\boldsymbol{W}) = \sum_{i=1}^{N} \parallel \boldsymbol{x}_i - \sum_{j=1}^{K} W_{i\Gamma_{ij}} \boldsymbol{x}_{\Gamma_{ij}} \parallel^2, \tag{1}$$

subject to the constraint $\sum_{j=1}^{K} W_{i\Gamma_{ij}} = 1$. It is easy to show that each weight can be calculated individually [8]. For each sample $\boldsymbol{x}_i$, construct a matrix $\boldsymbol{Q}$ with $Q_{jm} = \frac{1}{2}(\Delta_{i\Gamma_{ij}} + \Delta_{i\Gamma_{im}} - \Delta_{\Gamma_{ij}\Gamma_{im}})$. Let $\boldsymbol{R} = (\boldsymbol{Q} + r\boldsymbol{I})^{-1}$, where r is a suitably chosen regularisation constant (see [3]). Then $W_{i\Gamma_{ij}} = (\sum_{m=1}^{K} R_{jm}) / (\sum_{p,q=1}^{K} R_{pq})$.

In the second and final step, the weights stored in $\boldsymbol{W}$ are kept fixed and an embedding in $\mathbb{R}^M$ is found by minimising w.r.t. $\boldsymbol{Y}$:

$$\varepsilon_{II}(\boldsymbol{Y}) = \sum_{i=1}^{N} \parallel \boldsymbol{y}_i - \sum_{j=1}^{K} W_{i\Gamma_{ij}} \boldsymbol{y}_{\Gamma_{ij}} \parallel^2 . \tag{2}$$

This minimisation problem can be solved by introducing the constraint that the embedded data should have unit covariance, i.e. $\frac{1}{n}\boldsymbol{Y}\boldsymbol{Y}^T = \boldsymbol{I}$ (otherwise, $\boldsymbol{Y} = \boldsymbol{0}$ would minimise (2)). As a result, (2) is minimised by carrying out an eigen-decomposition of the matrix $\boldsymbol{M} = (\boldsymbol{I} - \boldsymbol{W})^T(\boldsymbol{I} - \boldsymbol{W})$ [8]. The eigenvectors corresponding to the 2^{nd} to $(M+1)^{st}$ smallest eigenvalues then form the final embedding $\boldsymbol{Y}$; the eigenvector corresponding to the smallest eigenvalue corresponds to the mean of the embedded data, and can be discarded to obtain an embedding centered at the origin.

After embedding, a new sample can be mapped quickly by calculating the weights for reconstructing it by its K nearest neighbours in the training set, as in the first step of LLE. Its embedding is then found by taking a weighted combination of the embeddings of these neighbours [3, 9].

LLE has been shown to be useful for analysis of high-dimensional data sets [3, 5, 6, 8]. A typical example is visualisation of a sequence of images, e.g. showing a person's face slowly rotating from left to right. For such data sets, LLE finds embeddings in which the individual axes correspond (roughly) to the small number of degrees of freedom present in the data.

2.2 Supervised LLE

Supervised LLE (SLLE, [3, 5]) was introduced to deal with data sets containing multiple (often disjoint) manifolds, corresponding to classes. For fully disjoint manifolds, the local neighbourhood of a sample $\boldsymbol{x}_i$ from class c $(1 \leq c \leq C)$ should be composed of samples belonging to the same class only. This can be achieved by artificially increasing the pre-calculated distances between samples belonging to different classes, but leaving them unchanged if samples are from the same class:

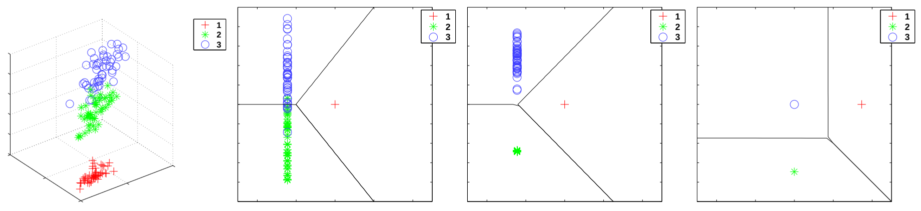
$$\boldsymbol{\Delta}' = \boldsymbol{\Delta} + \alpha \max(\boldsymbol{\Delta})\boldsymbol{\Lambda}, \quad \alpha \in [0, 1], \tag{3}$$

where $\Lambda_{ij} = 1$ if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ belong to the same class, and 0 otherwise. When $\alpha = 0$, one obtains unsupervised LLE; when $\alpha = 1$, the result is the fully supervised LLE introduced in [5] (called 1-SLLE). Varying α between 0 and 1 gives a partially supervised LLE (α-SLLE) [3].

For 1-SLLE, distances between samples in different classes will be as large as the maximum distance in the entire data set. This means neighbours of a sample in class c will always be picked from that same class. In practice, one therefore does not have to compute (3), but instead one can just select nearest neighbours for a certain sample from its class only. 1-SLLE is thereby a non-parameterised supervised LLE. In contrast, α-SLLE introduces an additional parameter α which controls the amount of supervision. For $0 < \alpha < 1$, a mapping is found which preserves some of the manifold structure but introduces separation between classes. This allows supervised data analysis, but may also lead to better generalisation than 1-SLLE on previously unseen samples.

2.3 Feature extraction

For $\alpha = 1$, the distance matrix represents C fully disconnected classes, each of which should be mapped fairly by LLE. These added degrees of freedom are used in the second step (2) to separate the classes, using the eigenvectors



(a) Original (b) LLE (c) 0.01-SLLE (d) 1-SLLE

Fig. 1. (a) First, third and fourth feature of the `iris` set ($N = 150$, $D = 4$, $C = 3$). (b)-(d) (S)LLE embeddings ($M = 2$), with trained nearest mean classifier.

corresponding to the 2^{nd} to C^{th} smallest eigenvalues, just as the first one was used to discard the mean of the data. Mapped classes are separated due to the constraint $\frac{1}{n}\boldsymbol{Y}\boldsymbol{Y}^T = \boldsymbol{I}$, and all samples in a certain class are mapped on a single point in $\mathbb{R}^{C-1}$. The optimal embedding dimensionality M therefore is $C-1$. For α-SLLE, this is not necessarily optimal, as this method is a trade-off between LLE and 1-SLLE. An automatic setting for M can be found by demanding that locally, on average, 90% of the variance is retained in the remaining M dimensions [3]. This local intrinsic dimensionality estimate is denoted by M_L.

The feature extraction process is illustrated in Figure 1: the $C = 3$ classes in the `iris` data set [1] are mapped onto single points by 1-SLLE. α-SLLE retains some of the class structure, but reduces within-class dispersion compared to LLE. Clearly, SLLE is suitable as a feature extraction step prior to classification. And, although internal structure of each class is (partially) lost during mapping, class overlap can easily be visualised in the $C - 1$ dimensional space.

The idea behind SLLE is related to that of spectral clustering [10]. There, first an affinity matrix between all samples is calculated. If clusters are present, this matrix will have a block-diagonal structure. An eigen-decomposition of the (normalised) affinity matrix then gives an embedding in a small number of dimensions, in which clusters are more clearly separated than in the original space. SLLE uses class label information to construct an artificial off-diagonal block matrix $\boldsymbol{\Lambda}$, and applies this to change the distance matrix used as the basis for LLE. The resulting matrix $\boldsymbol{W}$, which is already sparse (containing only K non-zero entries in each row), is changed towards a block-diagonal matrix. As a result, a mixture between unsupervised LLE and supervised spectral clustering is obtained.

3 Experiments

3.1 Setup

To verify the feature extraction capabilities of SLLE, it was applied to a number of data sets varying in number of samples N, dimensions D and classes C. The sets, together with the experimental results, are given in Tables 1 ($D \leq 30$) and 2

($D > 30$). Most of the sets were obtained from the UCI repository [1], some were used in our earlier work. The `chromosomes` set contains 30 gray-values sampled from chromosome banding profiles. The two `textures` sets contain 12×12-pixel gray-value image patches of either natural (i.e. unstructured) or structured (i.e. regular) Brodatz textures [2]. The `nist digits` set consists of 16×16-pixel gray-value images of pre-processed handwritten digits, taken from the NIST database [2]. Finally, the `paper` set contains 857 multiresolution local binary patterns, calculated on images of different types of paper [7].

The experiments were set up as follows: a set was randomly split into a training set (80%) and a test set (20%). Four classifiers were used: `nmc`, the nearest mean classifier; `ldc` and `qdc`, the Bayes plug-in linear and quadratic classifiers; and `knnc`, the K-NN classifier, with K optimised by the leave-one-out procedure on the training set [4]. This was repeated on data mapped by LLE and α-SLLE to M_L dimensions (see section 2.3) and by 1-SLLE to $C-1$ dimensions. Mappings were calculated for a range of values of K, the neighbourhood size parameter, and α (if applicable).

To compare the (S)LLE methods to more traditional feature extraction techniques, the classifiers were also trained on data mapped to M_L dimensions by PCA, LDA and Sammon mapping. PCA and LDA are linear methods, whereas Sammon mapping is nonlinear. LDA is the only supervised method of the three.

3.2 Results

Tables 1 and 2 present average errors on the test set (in %) over 10 trials, with the standard deviation given between brackets. The best result is shown in bold and underlined; all results falling within a standard deviation of the best result are shown in bold as well, as they are not significantly worse. For 1-SLLE and α-SLLE, only the best result found in the range of values for K and α found is shown. Ideally, these optimal values should be found on an independent validation set, but the size of many of the data sets did not permit setting aside samples for this.

The results confirm that SLLE generally leads to better classification performance than LLE and, usually, any other mapping technique. This is to be expected, as SLLE can extract nonlinear manifolds in a supervised way, and is thereby the most general of the feature extraction methods. Besides this, there are a number of interesting observations:

– **SLLE does not work on low-dimensional data.** For the low-dimensional sets shown in Table 1, the Bayes plug-in classifiers, `ldc` and `qdc`, often work well on the original data, as the number of parameters that need to be estimated is still reasonably low. In these cases, SLLE will not improve classification to the point where it is better than on the original data.
– **SLLE works well on high-dimensional data.** In some cases, performance is (nearly) as good as on the original data, but in others it is significantly better (**ionosphere**, **sonar**, the **textures** sets). The **splice** set is the exception: the quadratic classifier `qdc` performs surprisingly well and cannot be improved upon.

	Original	PCA	LDA	Sammon	LLE	1-SLLE	α-SLLE
iris [1]	$N = 150$, $C = 3$, $D = 4$, $M_G = 1$, $M_L = 3$						
`nmc`	7.7 (2.7)	9.3 (4.1)	**1.7** (2.4)	8.7 (4.5)	4.7 (3.9)	3.3 (3.1)	**1.7** (2.4)
`ldc`	**1.7** (2.4)	4.0 (3.1)	**1.7** (2.4)	4.7 (3.6)	$\underline{\mathbf{1.0}}$ (1.6)	12.0 (6.1)	$\underline{\mathbf{1.0}}$ (1.6)
`qdc`	3.3 (3.1)	4.3 (3.5)	3.3 (3.5)	5.0 (3.6)	**2.3** (2.2)	23.0 (27.7)	**2.3** (2.2)
`knnc`	**2.3** (1.6)	3.3 (3.1)	3.0 (3.3)	3.7 (1.9)	**2.3** (2.7)	3.3 (3.1)	**2.3** (2.2)
wine [1]	$N = 178$, $C = 3$, $D = 13$, $M_G = 1$, $M_L = 2$						
`nmc`	25.3 (5.3)	27.2 (7.9)	**1.1** (1.4)	27.2 (7.9)	25.3 (5.3)	4.7 (3.2)	5.8 (3.6)
`ldc`	**1.1** (1.4)	32.8 (6.1)	**1.1** (1.4)	34.2 (6.1)	26.7 (5.3)	41.1 (17.2)	15.6 (5.1)
`qdc`	$\underline{\mathbf{1.1}}$ (1.4)	27.8 (8.5)	**1.4** (1.5)	28.1 (8.1)	25.8 (6.4)	46.4 (11.0)	16.4 (5.1)
`knnc`	24.4 (5.4)	34.2 (6.3)	**1.1** (1.4)	34.7 (5.9)	24.7 (6.2)	4.7 (3.2)	11.4 (3.8)
diabetes [1]	$N = 768$, $C = 2$, $D = 8$, $M_G = 2$, $M_L = 4$						
`nmc`	34.5 (4.0)	39.2 (2.2)	24.2 (2.9)	34.5 (4.4)	27.1 (2.9)	25.1 (2.4)	25.5 (2.9)
`ldc`	$\underline{\mathbf{22.2}}$ (1.9)	34.5 (1.2)	**22.8** (2.3)	34.2 (1.2)	24.5 (1.8)	31.6 (4.6)	24.5 (1.8)
`qdc`	24.9 (1.9)	34.3 (2.0)	**22.5** (2.1)	34.5 (1.8)	28.1 (2.6)	35.1 (0.0)	27.5 (3.0)
`knnc`	24.4 (2.6)	32.9 (2.9)	**23.1** (2.5)	34.8 (1.5)	24.9 (2.9)	25.1 (2.4)	24.9 (2.9)
glass [1]	$N = 214$, $C = 6$, $D = 9$, $M_G = 4$, $M_L = 3$						
`nmc`	57.0 (6.8)	51.2 (5.3)	40.7 (9.8)	50.7 (5.1)	61.2 (5.7)	29.5 (4.8)	36.5 (6.5)
`ldc`	36.0 (5.7)	45.8 (11.0)	37.0 (10.0)	44.4 (8.9)	44.0 (8.1)	48.1 (6.2)	39.5 (7.4)
`qdc`	83.5 (7.2)	47.0 (4.5)	47.2 (5.9)	47.2 (6.8)	50.2 (6.5)	59.8 (5.3)	41.2 (6.1)
`knnc`	**28.4** (4.5)	$\underline{\mathbf{22.8}}$ (6.7)	38.1 (7.7)	**24.4** (8.3)	33.0 (7.0)	30.0 (4.7)	33.0 (7.0)
vehicle [1]	$N = 846$, $C = 4$, $D = 18$, $M_G = 1$, $M_L = 5$						
`nmc`	61.7 (1.7)	60.1 (1.6)	20.6 (2.4)	60.6 (2.1)	50.9 (3.4)	26.6 (4.6)	23.5 (3.3)
`ldc`	22.0 (3.9)	63.5 (3.9)	20.7 (2.6)	62.6 (2.9)	50.8 (3.4)	59.5 (1.3)	24.6 (3.7)
`qdc`	$\underline{\mathbf{14.4}}$ (2.1)	57.2 (2.6)	19.8 (1.9)	57.5 (4.0)	47.7 (2.9)	59.5 (1.3)	47.7 (2.9)
`knnc`	36.9 (2.8)	46.4 (2.3)	20.9 (2.2)	45.7 (2.0)	44.9 (3.7)	26.6 (4.6)	22.0 (3.2)
hepatitis [1]	$N = 80$, $C = 2$, $D = 19$, $M_G = 2$, $M_L = 3$						
`nmc`	**29.4** (9.8)	39.4 (10.6)	**29.4** (11.0)	38.8 (10.5)	**25.0** (10.6)	**29.4** (7.2)	**25.0** (10.6)
`ldc`	$\underline{\mathbf{22.5}}$ (13.9)	46.2 (6.0)	**29.4** (11.4)	45.6 (7.2)	**31.2** (5.1)	**33.8** (9.4)	**25.0** (13.5)
`qdc`	**32.5** (17.1)	46.2 (6.0)	**29.4** (11.4)	45.6 (6.6)	**30.0** (7.1)	**36.2** (9.2)	**25.6** (13.3)
`knnc`	39.4 (10.6)	48.1 (7.2)	**30.0** (14.7)	46.9 (6.8)	**34.4** (13.3)	**29.4** (7.2)	**24.4** (10.0)
chromosomes	$N = 2520$, $C = 24$, $D = 30$, $M_G = 8$, $M_L = 8$						
`nmc`	33.2 (2.0)	33.0 (1.8)	24.9 (1.4)	33.1 (1.7)	31.4 (1.7)	37.4 (1.5)	28.2 (1.7)
`ldc`	25.1 (2.5)	24.1 (1.0)	24.9 (1.4)	23.8 (1.0)	28.4 (1.0)	93.5 (0.5)	27.6 (1.8)
`qdc`	27.1 (1.4)	21.4 (1.3)	21.7 (1.9)	$\underline{\mathbf{19.3}}$ (1.7)	22.2 (2.2)	94.1 (1.4)	22.2 (2.2)
`knnc`	23.6 (1.7)	23.3 (1.5)	24.5 (1.9)	23.1 (1.2)	25.2 (2.0)	37.4 (1.5)	24.3 (1.7)

Table 1. Test error (in %), low-dimensional data sets.

- **1-SLLE vs. α-SLLE: neither consistently outperforms the other.** Although α-SLLE was expected to generalise better, there are two extra parameters to be estimated: α and the embedding dimensionality, M_L. The performance of α-SLLE is especially sensitive to the latter. Bayes plug-in classifiers trained on 1-SLLE mapped data perform poorly: as all samples are mapped onto a single point, there is no covariance structure left.
- **SLLE works well where K-NN works well on the original data.** SLLE is a neighbourhood-based method, like the K-NN classifier. In fact, SLLE can be seen as a generalised K-NN method, where not only the neighbours' labels play a role, but also the distances to these neighbours.

	Original	PCA	LDA	Sammon	LLE	1-SLLE	α-SLLE
ionosphere [1]				$N = 351$, $C = 2$, $D = 34$, $M_G = 18$, $M_L = 4$			
`nmc`	29.9 (5.5)	28.9 (7.6)	12.1 (2.7)	28.4 (7.6)	21.6 (3.8)	**7.7** (3.1)	**7.0** (2.5)
`ldc`	16.4 (7.2)	44.3 (5.9)	13.9 (2.9)	34.9 (6.9)	19.4 (4.5)	31.9 (5.7)	**7.0** (2.5)
`qdc`	11.4 (3.8)	38.0 (4.8)	12.6 (2.7)	35.4 (7.1)	19.0 (5.5)	26.9 (2.4)	**7.3** (2.1)
`knnc`	16.3 (3.1)	20.9 (3.4)	13.0 (2.7)	25.0 (5.1)	13.0 (2.2)	**7.7** (3.1)	**7.4** (1.8)
splice [1]				$N = 3188$, $C = 3$, $D = 60$, $M_G = 51$, $M_L = 18$			
`nmc`	23.9 (2.0)	44.1 (1.0)	21.3 (1.4)	37.6 (3.6)	36.2 (2.0)	19.5 (2.3)	17.2 (2.2)
`ldc`	19.0 (1.4)	33.2 (0.8)	19.3 (1.6)	30.9 (1.2)	35.6 (1.4)	75.9 (0.1)	17.6 (1.5)
`qdc`	**7.0** (1.2)	32.2 (1.2)	19.0 (1.3)	30.6 (1.6)	42.5 (1.8)	75.9 (0.1)	27.1 (1.9)
`knnc`	20.5 (1.4)	32.7 (1.2)	18.9 (1.5)	33.4 (4.0)	32.8 (2.0)	19.5 (2.3)	18.6 (2.1)
sonar [1]				$N = 208$, $C = 2$, $D = 60$, $M_G = 12$, $M_L = 8$			
`nmc`	32.4 (7.0)	46.8 (5.6)	25.4 (10.0)	46.8 (6.3)	23.4 (6.1)	**11.7** (3.0)	**13.7** (4.5)
`ldc`	25.4 (5.2)	44.6 (6.7)	25.6 (10.6)	45.4 (4.0)	22.0 (5.7)	53.7 (0.0)	**14.4** (4.8)
`qdc`	27.8 (5.9)	43.4 (5.6)	25.4 (10.9)	44.9 (4.9)	26.1 (5.4)	53.7 (0.0)	**14.6** (3.3)
`knnc`	18.5 (5.3)	51.7 (3.9)	24.6 (10.1)	49.3 (5.1)	18.8 (7.4)	**11.7** (3.0)	**12.9** (2.3)
optdigits [1]				$N = 5620$, $C = 10$, $D = 64$, $M_G = 21$, $M_L = 10$			
`nmc`	8.6 (0.7)	10.1 (0.8)	4.7 (0.7)	10.1 (0.6)	15.6 (1.5)	**1.4** (0.4)	**1.4** (0.4)
`ldc`	55.8 (44.3)	8.4 (1.0)	4.7 (0.7)	8.8 (0.6)	10.4 (1.2)	31.0 (1.0)	2.0 (0.4)
`qdc`	90.1 (0.0)	4.2 (0.5)	3.1 (0.5)	4.2 (0.4)	5.1 (1.0)	31.0 (1.0)	3.4 (0.6)
`knnc`	**1.2** (0.4)	2.8 (0.5)	2.9 (0.3)	3.0 (0.4)	3.2 (0.5)	**1.4** (0.4)	**1.3** (0.2)
natural textures [2]				$N = 3000$, $C = 6$, $D = 144$, $M_G = 33$, $M_L = 6$			
`nmc`	54.8 (2.1)	54.5 (1.2)	55.2 (1.7)	55.1 (1.7)	55.8 (1.9)	**30.6** (2.9)	**33.3** (2.6)
`ldc`	54.8 (1.5)	53.9 (1.4)	55.2 (1.7)	54.9 (1.5)	55.5 (1.6)	79.7 (0.9)	38.7 (2.3)
`qdc`	46.1 (1.8)	41.2 (2.2)	52.4 (2.1)	43.9 (2.0)	58.2 (3.0)	79.7 (0.9)	37.3 (2.2)
`knnc`	34.2 (1.0)	40.9 (2.7)	53.6 (1.4)	43.2 (2.2)	48.9 (2.5)	**30.6** (2.9)	**30.7** (2.9)
structured textures [2]				$N = 3000$, $C = 6$, $D = 144$, $M_G = 39$, $M_L = 17$			
`nmc`	51.3 (1.4)	52.0 (1.5)	55.8 (1.9)	52.7 (1.7)	28.1 (2.1)	9.5 (0.9)	**8.2** (0.8)
`ldc`	55.2 (1.7)	52.0 (1.5)	55.8 (1.9)	53.6 (1.2)	27.9 (1.6)	49.3 (1.9)	**8.3** (0.7)
`qdc`	24.3 (0.8)	30.2 (1.2)	50.4 (1.6)	36.3 (1.2)	13.5 (1.2)	49.3 (1.9)	11.7 (1.1)
`knnc`	13.8 (1.3)	30.0 (1.9)	52.9 (1.7)	37.6 (1.0)	14.5 (1.3)	9.5 (0.9)	**7.5** (0.9)
nist digits [2]				$N = 6250$, $C = 10$, $D = 256$, $M_G = 52$, $M_L = 12$			
`nmc`	16.5 (0.9)	21.5 (0.9)	10.3 (0.9)	20.1 (0.7)	17.1 (0.7)	3.4 (0.5)	**2.3** (0.3)
`ldc`	10.8 (0.4)	17.0 (1.0)	10.3 (0.9)	17.4 (0.7)	16.1 (0.8)	40.0 (1.1)	3.2 (0.7)
`qdc`	89.8 (0.4)	9.3 (1.0)	7.9 (0.8)	8.8 (0.9)	10.7 (0.6)	40.0 (1.1)	10.7 (0.6)
`knnc`	**2.5** (0.2)	6.7 (0.7)	7.0 (0.8)	7.1 (1.0)	6.7 (0.5)	3.4 (0.5)	**2.5** (0.4)
paper [7]				$N = 1004$, $C = 4$, $D = 857$, $M_G = 2$, $M_L = 7$			
`nmc`	3.3 (1.2)	2.8 (1.2)	26.7 (4.1)	1.4 (0.9)	0.3 (0.3)	**0.1** (0.2)	**0.2** (0.3)
`ldc`	75.1 (0.0)	1.2 (0.9)	75.1 (0.0)	**0.1** (0.2)	**0.2** (0.3)	2.9 (0.8)	**0.2** (0.3)
`qdc`	75.1 (0.0)	0.5 (0.5)	75.1 (0.0)	**0.1** (0.2)	21.3 (29.4)	2.9 (0.8)	21.3 (29.4)
`knnc`	**0.2** (0.3)	**0.1** (0.2)	26.7 (4.1)	**0.1** (0.2)	**0.2** (0.3)	**0.1** (0.2)	**0.1** (0.2)

Table 2. Test error (in %), high-dimensional data sets.

– **The nearest mean classifier performs well.** SLLE maps the data nonlinearly such that this simple classifier can do well. This is analogous to SVMs: after the kernel function performs the desired nonlinear mapping, a simple linear classifier suffices.

– **Local vs. global intrinsic dimensionality.** A priori one would expect SLLE to work well for sets which contain curved manifolds. Such sets would exhibit a high global intrinsic dimensionality coupled with a low local one. The tables show, besides the local intrinsic dimensionality estimates M_L, the number of PCA dimensions needed to preserve 90% of the variance globally, M_G. For all sets on which SLLE performs well (except **splice**), $M_L \ll M_G$.

The latter observation means that a simple test for the applicability of SLLE would be to quickly estimate M_L on a subset of samples and compare it to M_G. A significant difference indicates that good performance of SLLE is highly likely.

4 Conclusions

A common framework for unsupervised and supervised LLE was proposed. Experiments on a number of benchmark data sets demonstrated that SLLE is a powerful feature extraction method, which when coupled with simple classifiers can yield very promising recognition results. SLLE seems to be mainly applicable to high-dimensional data sets which clearly exhibit manifold structure.

Further research will address the problems of choosing α and M_L in a more well-founded way for α-SLLE. Computational and storage complexity is also still a concern: as the whole data set needs to be retained and is used in mapping new data, application to large data sets ($N = \mathcal{O}(10^4)$) is still infeasible. Subset selection [3] may alleviate this problem.

References

1. C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.
2. D. de Ridder. *Adaptive methods of image processing.* PhD thesis, Delft University of Technology, Delft, 2001.
3. D. de Ridder and R.P.W. Duin. Locally linear embedding for classification. Technical Report PH-2002-01, Pattern Recognition Group, Dept. of Imaging Science & Technology, Delft University of Technology, Delft, The Netherlands, 2002.
4. R.P.W. Duin. PRTools, a pattern recognition toolbox for Matlab, 2003.
5. O. Kouropteva, O. Okun, A. Hadid, M. Soriano, S. Marcos, and M. Pietikäinen. Beyond locally linear embedding algorithm. Technical Report MVG-01-2002, Machine Vision Group, University of Oulu, Finland, 2002.
6. O. Kouropteva, O. Okun, and M. Pietikäinen. Selection of the optimal parameter value for the locally linear embedding algorithm. In *Proc. of the 1st Int. Conf. on Fuzzy Systems and Knowledge Discovery, Singapore*, pages 359–363, 2002.
7. T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.
8. S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.
9. L.K. Saul and S.T. Roweis. Think globally, fit locally: unsupervised learning of nonlinear manifolds. Technical Report MS CIS-02-18, Univ. of Pennsylvania, 2002.
10. Y. Weiss. Segmentation using eigenvectors: a unifying view. In *Proc. of the IEEE Int. Conf. on Computer Vision (ICVV'99)*, pages 975–982, 1999.